GeoChemFoam User Guide

Julien Maes[1]

[1]Institute of GeoEnergy Engineering, School of Energy, Geosciences, Infrastructure and Society, Heriot-Watt University, Edinburgh, U.K.

1 Introduction

This document presents a description of GeoChemFoam, the Open Source OpenFOAM-based CFD solver developed at the Institute of GeoEnergy Engineering, Heriot-Watt University, Edinburgh, U.K. This code was developed with the objective of modelling subsurface processes at the micron-scale, but has also been applied to other multiphysics processes, such as mixing reactors.

This documentation does not intend to replace any OpenFOAM documentation, and the user should first familiarize themselves with the open-source toolbox (http://www.openfoam.com).

GeoChemFoam includes 11 packages that can model various multiphysics processes, from non-reactive multispecies transport in single-phase system to reactive multiphase transport. In this document, we give installation guide lines, a short description of the code packages and utilities and 10 tutorial test cases corresponding to each package.

2 Installation

The source code, applications and tutorial cases should be placed in the specified directory structure. The installation of this directory structure and instructions for how to compile the source codes is given in this section.

2.1 Prerequisite

GeoChemFoam-4.5 uses the community extended OpenFoam foam-extend-4.0. It is recommended to use the standard OpenFoam-4.x for meshing of complex domain such as micro-CT images. The mesh can then be converted to foam-extend format using the compactFaceToFace utility. Installation instruction for these package can be obtained at:

https://openfoamwiki.net/index.php/Installation/Linux/foam-extend-4.0.

https://openfoamwiki.net/index.php/Installation/Linux/OpenFoam-4.x.

https://github.com/petebachant/compactFaceToFace

Paraview, available with any OpenFOAM distribution or through the package manager of most popular Linux distributions, can be used for visualizing and also post processing the results.

2.2 Set-up files directory

Create a directory in your home folder named 'works'.

```
mkdir $HOME/works
cd $HOME/works
```

Extract the GeoChemFoam folder here under the name 'GeoChemFoam-4.5'.

2.3 Set-up the bashrc script

The bashrc file gives the path to the foam-extend-4.0 directory, as well as a few aliases that are used during compilation. It is localted in the 'etc' directory. To set-it up to your system, change line #10, #11 and #12 according to your foam-extend-4.0 and OpenFoam-4.x installation directories. Then source the bashrc file.

```
source etc/bashrc
```

2.4 Compilling

First, compile PhreeqcRM, the reaction module, that can be found in the ThirdParty folder:

```
cd ThirdParty
./Allwmake
cd ..
```

Then, type the following command to compile the code:

```
./Allwmake
```

2.5 Checks

To check that the code has correctly compiled, one can run the checkInstall command:

```
./checkInstall.sh
```

If the code has compiled correctly, the final line should be:

```
Installation successful
```

In addition, each package can be tested through the tutorials presented in the section 5.

3 Packages

GeoChemFoam-4.5 includes 11 different packages that allow for resolution of various multi-physic problems, from non-reactive single-phase transport to multicomponent multiphase reactive transport. Each of these packages is a separate solver that can be found in the 'applications/solvers' directory. Definition of objects and structures used in these solver and specific to GeoChemFoam, such as 'interfaceProperties' and 'reactionModule' can be found in the 'src' directory. The reactive part is solved using the USGS geochemical solver Phreeqc https://wwwbrr.cr.usgs.gov/projects. Each of the 11 solvers is given at least one tutorial test case.

3.1 multiSpeciesTransportFoam

The multiSpeciesTransportFoam package is located in in 'applications/solvers/transport'. It is a modified version of the standard scalarTransportFoam OpenFoam package to allow for multispecies transport in solution. The species in solution are defined in the 'constant/thermoPhysicalProperties' file, using the keyword solutionSpecies, and the diffusion coefficient D is given. Example:

```
solutionSpecies
{
    Na+
    {
        D D [0 2 -1 0 0 0 0] 1e-9;
    }
    Cl-
    {
        D D [0 2 -1 0 0 0 0] 2e-9;
    }
};
```

The solver for the species concentration Yi needs to be defined in the 'system/fvSolution' file. Example:

```
Yi
{
    solver              PBiCG;
    preconditioner      DILU;
    tolerance           1e-7;
    relTol              0;
}
```

Finally, the discretization scheme for the flux of Yi needs to be defined in the 'system/fvSchemes' file. Example:

```
divSchemes
{
    div(phi,Yi)   Gauss Gamma 0.5;
}
```

3.2 reactiveTransportFoam

The reactiveTransportFoam package is located in the 'applications/solvers/reactiveTransport' folder. It extends the multiSpeciesTransportFoam package to reactive transport using the chemical reaction package Phreeqc https://wwwbrr.cr.usgs.gov/projects. The chemical reactions supported by the GeoChemFoam's Phreeqc module are aqueous equilibrium and surface complexation. Mineral dissolution is not handle by Phreeqc but taken care of using boundary conditions (see the dissolTransportDyMFoam package). A Phreeqc database needs to be present in the 'constant/GeoChem.dat' file along with an additional file called 'phreeqcReactions', which defined the reactions included in the phreeqc model, and a default solution. Example:

```
SURFACE_MASTER_SPECIES
Surf_w  Surf_w

SOLUTION_MASTER_SPECIES
T T 0 1 1
A A 0 1 1

SOLUTION_SPECIES
T = T
A = A
A + T = AT
-log_k 30

SURFACE_SPECIES
Surf_w = Surf_w
Surf_w + T = Surf_wT
-log_k -104

SOLUTION 0
units Mol/L
T 1e-3
A 1e-3
```

Surface species and surface master species can be defined in the 'constant/thermoPhysicalProperties' file, by declaring their name. The definition of surface master components can be found in the Phreeqc documentation present in 'ThirdParty/docs'. Surface species need to start by the prefix 'Surf' for GeoChemFoam to recognize them. Example:

```
surfaceSpecies
{
    Surf_w;
    Surf_wT;
};
surfaceMasters
{
    Surf_w;
};
```

Moreover, the surface masters must be defined in the '0/Surf' file. To define the boundary surfaces and which surface reactions occurs, GeoChemFoam uses a new boundary condition called reactingWall, located in 'src/boundaryConditions/reactingWall'. The boundary conditions requires the site density of each master species. Example:

```
FoamFile
{
    version 4.0;
    format ascii;
    class volScalarField;
    Object Surf;
}

dimensions [0 -2 0 0 1 0 0];
internalField uniform 0;

walls
{
    type reactingWall;
    surfaceMasters
    {
        Surf_w
        {
            density 8e-6;
        }
    }
    value uniform 8e-6;//[mol/m2]
}
```

3.3 dissolTransportDyMFoam

The dissolTransportDyMFoam package is located in the 'applications/solvers/reactiveTransport' folder. This package is designed to perform mineral dissolution by acid injection. Unlike *reactiveTransportFoam*, the chemical reaction is not handled by Phreeqc. Instead the mineral dissolution is modelled using a simplified reaction:

$$\text{Mineral} + \text{Acid} \rightarrow \text{liquid} \tag{1}$$

The reaction is handled as a boundary condition for the acid concentration on the solid surface using the *reactiveSurfaceConcentrationMixed* boundary condition in which the reaction constant k (kmol/m^2/s) is defined. Example:

```
  movingWalls
  {
      type            reactiveSurfaceConcentrationMixed;
      k               8.9125e-4; //kmol/m2/s;
      value           uniform 0;
  }
```

The displacement of the solid/fluid interface is handled by moving the mesh using an Arbitrary Eulerian Lagrangian (ALE) method [1]. For each time-step, the mesh is moved using the openFoam *dynamicMesh* feature. The numerical method used to deform the mesh is defined in the 'constant/dynamicMeshDict' file. We are using a laplacian equation to displace the mesh points while satisfying the velocity at the moving boundary. Example:

```
FoamFile
{
    version 4.0;
    format ascii;
    class dictionary;
    location "constant";
    Object dynamicMeshDict;
}

dynamicFvMesh dynamicMotionSolverFvMesh;
motionSolverLibs ( "libfvMotionSolver.so" );
solver velocityLaplacian;
diffusivity quadratic inverseDistance (movingBoundaryName);
```

The discretization scheme for the laplacian of the cell motion needs to be defined in the 'system/fvSchemes' file. Example:

```
laplacianSchemes
{
    laplacian(diffusivity,cellMotionU)  Gauss linear uncorrected;
}
```

In addition, the '0' folder must contain a file 'pointMotionU', which define the mineral surface that will be displaced by the dissolution. The reacting surface must be of the boundary type 'dissolSolidVelocity', while fixed surface must be of the type 'uniformedFixedValue'. The 'dissolSolidVelocity' boundary condition requires the reaction constant k (kmol/m^2/s), the same as in the acid concentration boundary condition, the density of the solid phase rhos (kg/m^3) and the molecular weight of the phase component Mw (kg/kmol). Example:

```
walls
{
    type dissolSolidVelocity;
    k    8.9125e-4;//kmol/m2/s
    Mw   100;//kg/kmol
    rhos 2700;//kg/m3
    value uniform (0 0 0);
    uniformValue (0 0 0);
}

fixedWalls
{
    type uniformFixedValue;
    value uniform (0 0 0);
    uniformValue (0 0 0);
}
```

For each time-step, the mesh is displaced first, using the solver for the cell motion defined in the 'system/fvSolution' file. Example:

```
cellMotionU
{
    solver              PBiCG;
    preconditioner      DILU;
    tolerance           1e-7;
    relTol              0;
}
```

Then, steady-state flow and transport are solved. Parameters for the steady-state schemes need to be defined in the 'system/fvSolution' file, including the number of non-orthogonal correctors (nNonOrthogonalCorrectors), the maximum number of iterations (nCorrectors) and the convergence criteria for each variable (residualControl).

```
STEADYSTATE
{
    nNonOrthogonalCorrectors 1;
    nCorrectors 2000;

        residualControl
        {
                p                   1e-6;
                U                   1e-6;
                H+                  1e-6;
        }
}
```

Finally, in the 'system/controlDict' file, the library fvMotionSolverGCFOAM (for the dissolSolidVelocity boundary condition) should be specified, and the parallel communication type should be set to blocking, to avoid an error related to the mesh motion:

```
FoamFile
{
    version 4.0;
    format ascii;
    class dictionary;
    location "system";
    Object controlDict;
}

libs ("libfvMotionSolverGCFOAM.so");
OptimisationSwitches
{
    commsType      blocking;
}
```

3.4 interGCFoam

The interGCFoam package is located in 'applications/solvers/multiphase'. It is a modified version of the standard interFoam package for two-phase incompressible flow simulation with the VOF method https://openfoamwiki.net/index.php/InterFoam.

Standard interFoam uses the traditional Continuous Surface Force (CSF) method [2]. CSF has been reported to generate additional non-physical spurious velocities [3]. Our version of interFoam includes the following modifications to try to mitigate spurious velocities:

- Smoothing of interface indicator function α for computation of interface curvature,

$$\begin{aligned} &\alpha_{s,0} = \alpha, \\ &\alpha_{s,i+1} = cSK \left\langle \langle \alpha_{s,i} \rangle_{c \to f} \right\rangle_{f \to c} + (1 - cSK)\, \alpha_{s,i}. \end{aligned} \tag{2}$$

  This gives a more accurate computation of interface normal vector at cell centers [4].

- Computation of curvature from interface normal vector at cell center,

$$(n_I)_f = \langle n_I \rangle_{c \to f} = \left\langle \frac{\nabla \alpha}{|\nabla \alpha|} \right\rangle_{c \to f}. \tag{3}$$

  This improves the computation of the interface curvature [5].

- Complex expression for curvature

$$\kappa = -\nabla (n_I)_f + n_I \nabla (n_I)_f \, n_I. \tag{4}$$

  The correction insures that the interface vector remains of unit norm during computation of the interface curvature [5].

- Sharpening of indicator function α for computation of surface tension force,

$$\begin{aligned} &f_{c,f} = \sigma \langle \kappa \rangle_{c \to f} \, \delta_{pc}, \\ &\delta_{pc} = \nabla \alpha_{pc}, \\ &\alpha_{pc} = \frac{1}{1 - cPc} \left[\min \left(max \left(\alpha, \frac{cPc}{2} \right), 1 - \frac{cPc}{2} \right) - \frac{cPc}{2} \right]. \end{aligned} \tag{5}$$

  This allows for a sharper representation of the surface tension force and eliminate parasitic velocities due to numerical diffusion when $\alpha \approx 0$ and $\alpha \approx 1$ [4].

To run interGCFoam, the following parameters need to be defined in the 'case/system/fvSolution' file, in the PIMPLE section:

- Smoothing constant cSK.
- Number of smoothing iterations nSK.
- Sharpening parameter cPc.

Although each of the modification and correction may be beneficial, applying a too large value for the coefficients can decoralate the surface tension force to the global dynamic of the system and create problems such as interface smearing and velocity bursts. Therefore, we recommend to use the following the following set of parameters:

```
cSK         0.5;
nSK         1;
cPc         0.1;
```

Moreover, the capillary pressure solver needs to be defined in the 'system/fvSolution' file. Example:

```
pc
{
    solver            PCG;
    preconditioner    DIC;
    tolerance         1e-7;
    relTol            0;
}
```

3.5 interOSFoam

The interOSFoam package is located in 'applications/solvers/multiphase'. It is a modified version of interGCFoam that uses the Operator Splitting with Capillary Relaxation (OSCAR) method [14]. interOSFoam can be uses at low capillary number $Ca \leq 10^{-5}$ to solve the equation with larger time-steps (CFL$\approx$ 0.1). For one time-step, a large number of capillary relaxation steps are necessary, but they might converge before reaching there last iteration, leading to a large speed-up.

In addition to all keywords necessary to run interGCFoam, interOSFoam also requires to provide three additional parameters in the PIMPLE dictionary of 'system/fvSolution': the capillary relaxation time-step deltaTCR (this should be lower than the Brackill time-step), the minimum number of relaxation steps and the residual for convergence of the relaxation steps.

```
PIMPLE
{
    pcRefCell 0;
    pcRefValue 0;
    cSK 0;
    nSK 0;
    cPc  0;

    momentumPredictor no;
    nOuterCorrectors 1;
    nCorrectors      3;
    nNonOrthogonalCorrectors 0;
    nAlphaCorr       1;
    nAlphaSubCycles 1;

    deltaTCR 5e-8;
    relaxMin 4;
    residual 1e-4;
}
```

3.6 interTransportFoam

The interTransportFoam package is located in the 'applications/solvers/multiphaseTransport' folder. It merges the interFoam and the multiSpeciesTransport packages. Therefore, it requires all keywords necessary to run both applications. Moreover, the diffusion coefficients in both phases, the Henry's constant and the molecular weight for each species need to be defined in the 'constant/thermoPhysicalProperties' file. Example:

```
solutionSpecies
{
    CO2
    {
        D1 D1 [0 2 -1 0 0 0 0] 1.4e-5;
        D2 D2 [0 2 -1 0 0 0 0] 2e-9;
        H  H  [0 0 0 0 0 0 0]  0.8;
        Mw Mw [1 0 0 0 -1 0 0] 0.044;
    }
    H2
    {
        D1 D1 [0 2 -1 0 0 0 0] 1e-5;
        D2 D2 [0 2 -1 0 0 0 0] 4e-9;
        H  H  [0 0 0 0 0 0 0]  0.02;
        Mw Mw [1 0 0 0 -1 0 0] 0.002;
    }
};
```

The multiphase mass conservation equation for each species solved using the Compressive Continuous Species Transfer (CCST) method [6, 7]:

$$\begin{aligned} &\frac{\partial Y_i}{\partial t} + \nabla.\left(Y_i \mathbf{u}\right) + \nabla \cdot \left(c_{Y_i} \frac{\left(1-H_i\right) Y_i}{\alpha_1 + H_i \alpha_2} \alpha_1 \alpha_2 \mathbf{u}_r \right) = \nabla. \left(\hat{D}_i \nabla Y_i - \hat{D}_i \frac{\left(1-H_i\right) Y_i}{\alpha_1 + H_i \alpha_2} \nabla \alpha_1 \right), \\ &\hat{D}_i = \frac{\alpha_1 D_{i,1} + H_i \alpha_2 D_{i,2}}{\alpha_1 + H_i \alpha_2}. \end{aligned} \tag{6}$$

As shown in Maes et al. (2020) [7], it is essential to solve the advection part of the transport equation using exactly the same discretization scheme that for the volume fraction equation. Therefore, in *interTransportFoam*, Equ. 6 is solved using a sequential operator splitting method with the advection solved first separately using MULES and then injected as a source term in the diffusion equation. The constant c_{Y_i} is defined in the PISO dictionary in the 'system/fvSolution' file. It is recommended to use the same value for c_{Yi} to c_α to keep the coherence between the advection of Y_i and α. Example:

```
PISO
{
    cAlpha  1.0;
    cYi     1.0;
}
```

In two-phase transport, the boundary conditions for the global composition of a species at a wetted wall is not zeroGradient, even when it is for the partial composition in each phase. Instead, the gradient of the global composition must be calculated from the gradient of α, so that [8]:

$$\nabla Y_i = \frac{\left(1-H_i\right) Y_i}{\alpha_1 + H_i \alpha_2} \nabla \alpha_i \tag{7}$$

This is defined by the use of the 'globalConcentrationMixed' boundary composition for a species on a wetted wall. Example:

```
dimensions [0 -3 0 0 1 0 0];
internalField uniform 0;

boundaryField
{
    walls
    {
        type    globalConcentrationMixed;
    }
}
```

To use this boundary condition, the following line should be included in the 'system/controlDict' file:

```
libs ("libreactionThermoPhysicalModelGCFOAM.so");
```

The discretization scheme for the solubility flux div(phiH,Yi) needs to be defined in the 'system/fvSolution' file. We recommend to use the Gauss linear scheme, which seems to give the best results. Example:

```
divSchemes
{
    div(rho*phi,U)      Gauss SFCD
    div(phi,alpha)      Gauss vanLeer;
    div(phi,Yi)         Gauss vanLeer;
    div(phirb,alpha)    Gauss interfaceCompression;
    div(phiH,Yi)        Gauss upwind;//for upwind by phase scheme
}
```

3.7 interTransferFoam

The interTransfertFoam package is located in the 'applications/solvers/multiphaseTransport' folder. It is an extension of the interTransportFoam solver that takes into account local volume change induced by interface mass transfer [7]. Here, phase 1 is the continuous phase and phase 2 is the discontinuous phase. The discontinuous phase is formed of species that are soluble in the continuous phase. The continuous phase is formed of the soluble species plus a solvent which is not soluble in the discontinuous phase. It is important for mass conservation that test cases are set-up with the total mass concentration of all species in phase 2 equal to the mass density of the phase [7].

3.8 interReactiveTransportFoam

The interReactiveTransportFoam package is located in the 'applications/solvers/multiphaseReactiveTransport' folder. It merges the interTransportFoam and the reactiveTransportFoam package, therefore it required all keywords necessary to run these two applications. The model is described in [13]. Moreover, it is essential to define which phase will be modelled using Phreeqc in the 'constant/thermoPhysicalProperties' file. Example:

```
Phase1 phreeqcMixture;
Phase2 inertMultiComponentMixture;

solutionSpecies
{
        H+
        {
                D1 D1 [ 0 2 -1 0 0 0 0 ] 1e-9;
                D2 D2 [ 0 2 -1 0 0 0 0 ] 1e-9;
                H H [ 0 0 0 0 0 0 0 ] 1e-4;
                Mw Mw [1 0 0 0 -1 0 0] 1;
        }


        OH-
        {
                D1 D1 [ 0 2 -1 0 0 0 0 ] 1e-9;
                D2 D2 [ 0 2 -1 0 0 0 0 ] 1e-9;
                H H [ 0 0 0 0 0 0 0 ] 1e-4;
                Mw Mw [1 0 0 0 -1 0 0] 17;
        }
};
```

3.9 simpleDBSFoam

This package is an adaptation of the standard *simpleFoam* OpenFOAM package that solves the Darcy-Brinkman-Stokes Equation:

$$\frac{1}{\epsilon}\left(\frac{\mathbf{u}}{\epsilon}\mathbf{u}\right) = \frac{\nu}{\epsilon}\nabla^2 U - \nabla p - \nu k^{-1} u \tag{8}$$

where ν (m^2/s) is the the kinematic viscosity, ϵ is the porosity and k is the permeability. This package is especially design to model the impact of subresolution porosity in micro-CT images [9]. In addition to all parameters required to run the standard *simpleFoam* package, two additional fields called eps and Kinv representing the porosity and the inverse of permeability need to be present in the '0' folder. If Kinv is not present in the '0' folder, it will be calculated using Kozeny-Carman equation

$$k^{-1} = kf\frac{(1-\epsilon)^2}{\epsilon^3}, \tag{9}$$

where kf is a constant given in the 'constant/transportProperties' file. Example:

```
kf kf [0 -2 0 0 0 0 0] 1e+12
```

3.10 heatTransportSimpleFoam

This package is an adaptation of the basic *laplacianFoam* OpenFOAM package in order to solve the simplified steady-state temperature equation [15]

$$\nabla \cdot (\mathbf{u}T) = \nabla D_T \nabla T \tag{10}$$

where D_T is the local thermal diffusivity. Both solid and fluid can be present in the domain and the local porosity eps needs to be present in the '0' folder. The thermal diffusivity is then calculated using harmonic averaging

$$D_T = \frac{D_{Tf} D_{Ts}}{\epsilon D_{Ts} + (1-\epsilon) D_{Tf}}, \tag{11}$$

where D_{Tf} and D_{Ts} are the fluid and solid thermal diffusivity, given in the 'constant/transportProperties' file. Example:

```
DTf DTf [0 2 1 0 0 0 0] 1.4e-7
DTs DTs [0 2 1 0 0 0 0] 1.4e-6
```

In addition, the linear solver for the temperature equation needs to be given in the 'fvSolution'. Example:

```
T
{
    solver          PBiCG;
    preconditioner  DILU;
    tolerance       1e-7;
    relTol          0;
}
```

Finally, the discretization scheme for the flux of T needs to be defined in the 'system/fvSchemes' file. Example:

```
divSchemes
{
    div(phi,T)  Gauss upwind;
}
```

The equation is then solved using the SIMPLE algorithm as used in the basic *laplacianFoam* solver.

4 Pre-processing and post-processing utilities

In addition to the 11 packages present in the 'applications/solvers' directory, the 'applications/utilities/' directory includes 9 pre- and post- processing package that are useful for the preparation interpretation of the solver results. Each of these post-processing utility use the 'system/postProcessDict' to defined the bounding bbox $((x_1 y_1 z_1)(x_2 y_2 z_2))$ on which the utility is applied. Example:

```
FoamFile
{
version     4.0;
format      ascii;
class       dictionary;
location    "system";
object      postProcessDict;
}

x1 0;
y1 0;
z1 0;

x2 0.01;
y2 0.01;
z2 0.01;

direction 0;
```

4.1 processPoroPerm

This package calculates the porosity and the permeability of the domain at each time-step. This is useful to calculate the permeability of a micro-CT image after running *simpleFoam* or *simpleDBSFoam*, or to monitor the change of porosity and permeability of a image due to dissolution. The porosity is calculated as the fraction of volume of the computational domain V, weighted by eps in case of a DBS simulation, to the volume of the bounding box V_{bb}.

$$\phi = \frac{1}{V_{bb}} \int_V \epsilon dV \tag{12}$$

The permeability is calculated as:

$$K_j = \frac{U_j \nu}{\frac{\partial p}{\partial x_j}} \tag{13}$$

where U_j is the average velocity in the direction j in the bounding box

$$U_j = \frac{1}{V_{bb}} \int_V u_j dV \tag{14}$$

The pressure drop can be calculated as the rate of energy dissipation due to the internal viscous force per unit of flow rate [11],

$$\frac{\partial p}{\partial x_j} = \frac{1}{U_j V_{bb}} \int_V \nu \nabla^2 u \cdot u dV \tag{15}$$

The direction of the flow should be define in the 'system/postProcessDict' file. The permeability is calculated for the bounding box defined in 'system/postProcessDict'.

4.2 processConcentration

This package calculates the average concentration of each species in a bounding box, defined in the 'system/postProcessDict' file. If no bounding box is defined, then the concentration is calculated for the full domain .

4.3 processMaxVelocity

This package writes in a 'maxVelocity.csv' file the maximum velocity for each report time. This is useful to monitor the rise of parasitic current in multiphase simulations.

4.4 processSaturation

This package calculate the saturation of each phase. The saturations are calculated in a bounding box, defined in the 'system/postProcessDict' file. If no bounding box is defined, then the saturations are calculated for the full domain .

4.5 processPhaseConcentration

This package calculates the average concentration of each species in each phase. The concentration are calculated in a bounding box, defined in the 'system/postProcessDict' file. If no bounding box is defined, then the concentrations are calculated for the full domain.

4.6 processInterfaceTransfer

This package calculates the total interfacial transfer per unit of interfacial area (kg/m^2/s). The transfer is calculated in a bounding box, defined in the 'system/postProcessDict' file. If no bounding box is defined, then the transfer is calculated for the full domain.

4.7 processHeatTransfer

This package calculates the heat exchange coefficient (m/s) between the solid and the fluid. The transfer is calculated in a bounding box, defined in the 'system/postProcessDict' file. If no bounding box is defined, then the transfer is calculated for the full domain.

4.8 processMeshCellCenters

This package construct a vectorField '0/cellcenters' that provide the coordinate of the center of each grid block.

4.9 smoothSolidSurface

This package is used to smooth the fluid-solid interface by applying a laplacian smoother to the local porosity field '0/eps'.

$$\begin{aligned} &\epsilon_{s,0} = \epsilon, \\ &\epsilon_{s,i+1} = cSK \left\langle \langle \epsilon_{s,i} \rangle_{c \to f} \right\rangle_{f \to c} + (1 - cSK)\, \epsilon_{s,i}. \end{aligned} \tag{16}$$

And we need to provide the number of smoothing iterations and the smoothing constant nSK in the 'system/fvSolution' file in the 'Simple' directory. Example:

```
cSK             0.5;
nSK             1;
```

The textitsmoothSolidSurface applications update the '0/eps' field with the smoothed one.

5 Tutorials

5.1 Multi-species transport in Ketton micro-CT image

This test case simulate the transport of components in a Ketton micro-CT image. Two components are considered, Na^+, with a diffusion coefficient D=10^{-8} m^2/s, and Cl^-, with a diffusion coefficient D=10^{-11} m^2. The aim is to compare a diffusion dominated transport and an advection dominated transport.

First, the domain is defined and meshed using the 'initCase.sh' script

```
./initCase.sh
```

A 512^3 voxel raw micro-CT image with resolution 5.3 micron is included in 'constant/triSurface'. The 'raw2stl.py' python script in the same folder create the corresponding stl image. A 300^3 box is meshed using a 75×75×75 mesh is created. The pore-space is then meshed using the OpenFOAM *snappyHexMesh* utility using refinementSurface, so that each cells crossed by a solid face is refined twice in each direction (making the resolution if the mesh 10.6 microns).

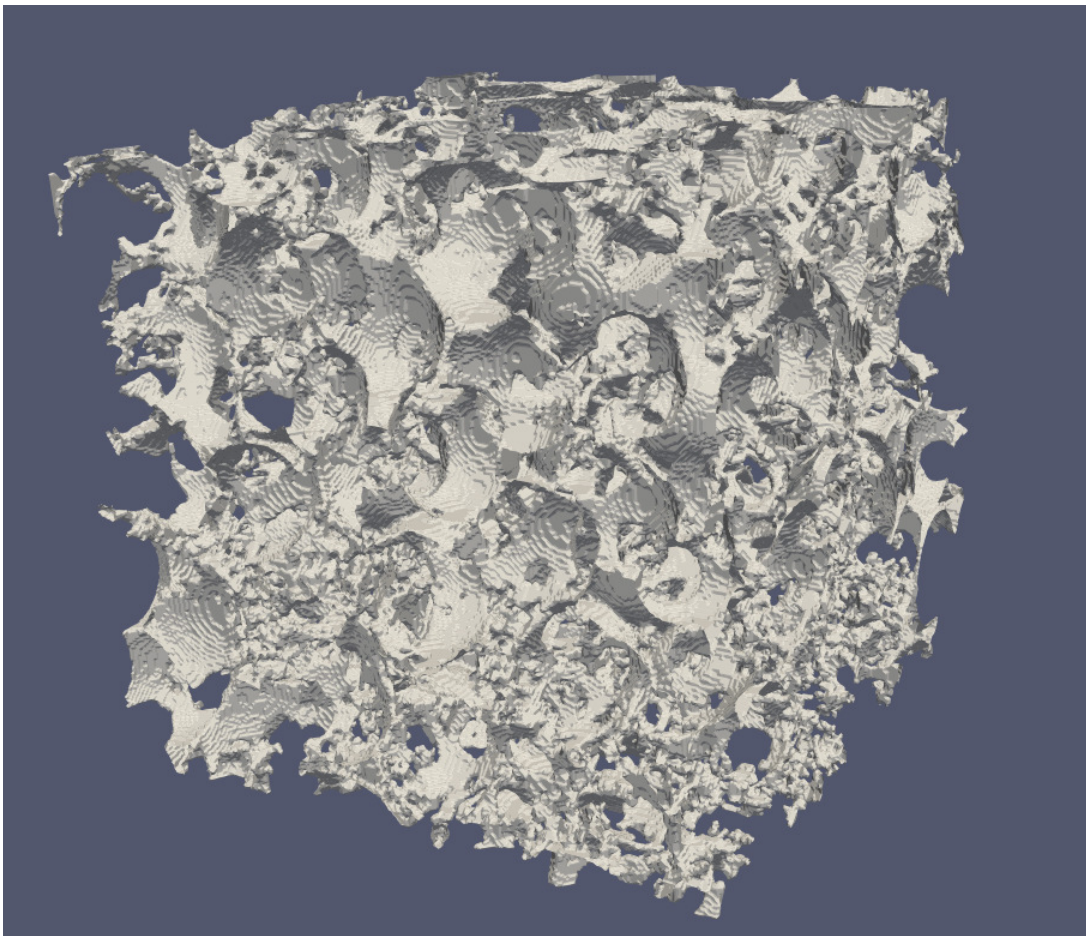

Figure 1: Ketton micro-CT image

Then, the script 'runCaseFlow.sh' calculate the flow field using the OpenFOAM *simpleFoam* solver, and then calculate the porosity and permeability using the processPoroPerm utility. The results are written in a 'poroPerm.csv' file.

```
./runCaseFlow.sh
```

Finally, the script 'runCaseTransport.sh' solve transport of Na^+ and Cl^- from 0 to 900s, and the average concentration in the domain are calculated and written in 'Na+_conc.csv' and 'Cl-_conc.csv'.

```
./runCaseTransport.sh
```

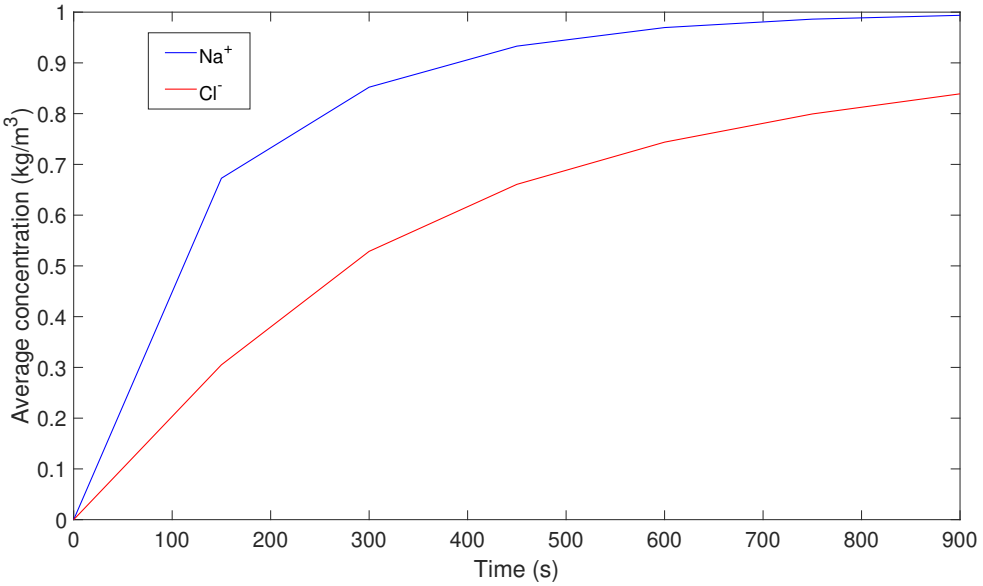

Figure 2: Evolution of average concentration of two components in a Ketton micro-CT image

5.2 Bimolecular reaction in a grainstone micromodel

This case simulate the instantaneous homogeneous bimolecular reaction A+T→AT in a 2D grainstone micromodel. The chemical system is described in 'constant/phreeqcReactions'. The geometry is defined in an stl image in 'constant/triSurface', and the pore-space is meshed using the OpenFOAM *snappyHexMesh* bu running the 'initCase.sh' script.

```
./initCase.sh
```

Initially, the domain is filled with species A with concentration c_0=1 kmol/m^3. At t=0, we introduce from the left boundary the species T with concentration c_0 and constant velocity u= 0.1 mm/s. The diffusion coefficient of all species in the domain is equal to $D = 10^{-9}$ m^2/s.

Then, the script 'runCaseFlow.sh' calculate the flow field using the OpenFOAM *simpleFoam* solver, and then calculate the porosity and permeability using the processPoroPerm utility. The results are written in a 'poroPerm.csv' file.

```
./runCaseFlow.sh
```

Finally, the script 'runCaseReactiveTransport.sh' solve the reactive transport problem from 0 to 4 s, and the average concentration in the domain are calculated and written in 'A_conc.csv', 'T_conc.csv' and 'AT_conc.csv'

```
./runCaseReactiveTransport.sh
```

Fig. 3 shows the concentration of AT at t=4 s.

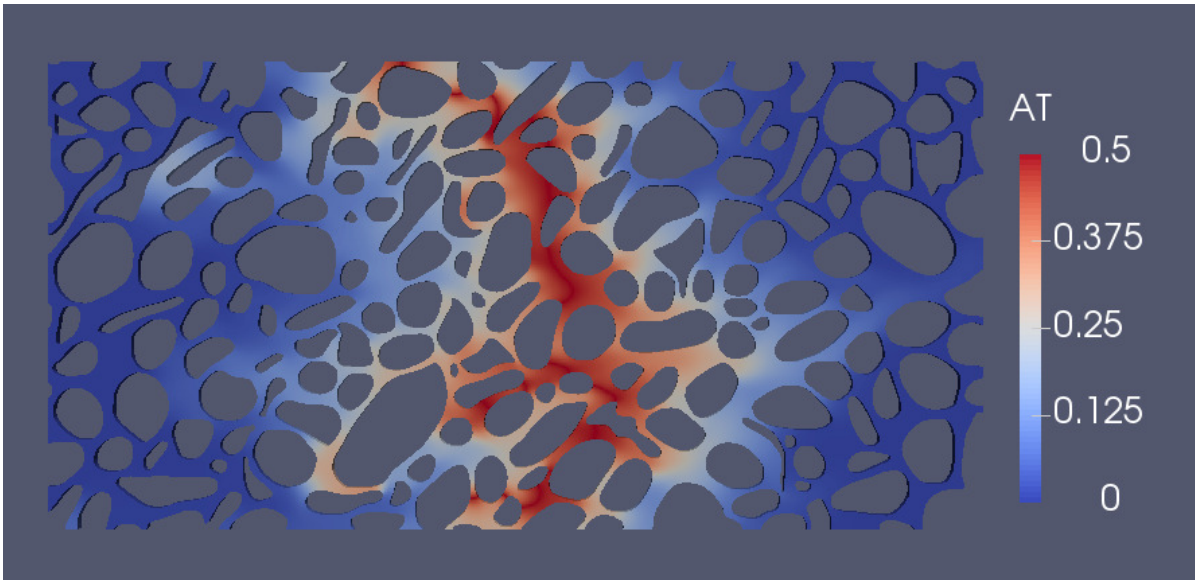

Figure 3: Concentration of product AT of bimolecular reaction during reactive transport in 2D grainstone micromodel

5.3 3D calcite post

This test case simulates the dissolution of a calcite post by injection of acidic water in a microchannel. The geometry and fluid properties are described in [10].

The script 'runCase.sh' run the simulation until t=1800 s and calculate the evolution of porosity and permeability using the processPoroPerm utility.

```
./runCase.sh
```

The evolution of the mineral shape and acid concentration can be observed using paraview (see Fig. 4). We observe that the mesh deforms as the calcite dissolves.

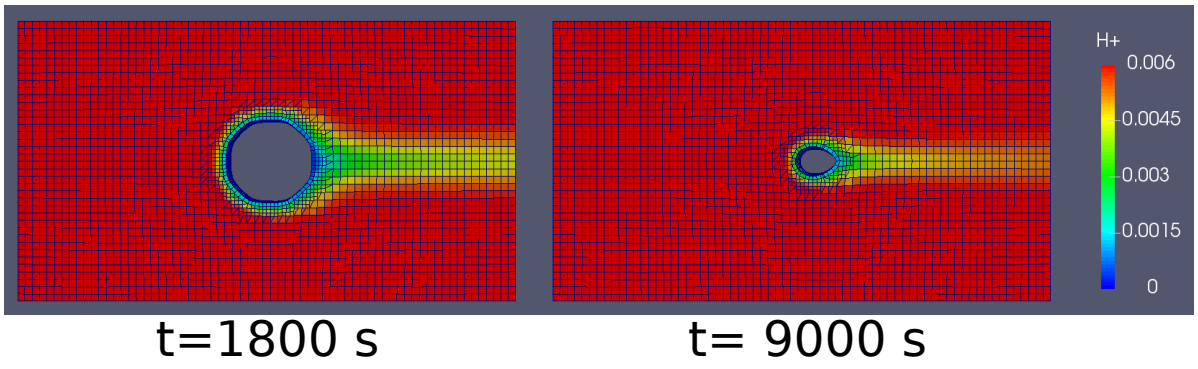

Figure 4: Numerical simulation of the dissolution of a calcite post in acidic water

5.4 2D throat

This test case is a benchmark case for evaluation of spurious velocities in a dynamic regime. We consider a 2D pore throat, of dimension 100μm×40μm. The two fluids are water (ρ = 1000 kg/m^3, μ= 1 mPa.s) and oil (ρ = 750 kg/m^3, μ=1.5 mPa.s). The surface tension is set equal to 50 mN/m. We use a uniform 50×20 uniform grid. The maximum Brackbill time-step

is equal to 0.21 μs, so we use a maximum time-step of 0.2 μs. The pore walls are set to be oil-wet with a contact angle of 30^o. The domain is initially filled with 20% water and 80% oil and we inject water from the left boundary with a constant velocity equal 0.5 mm/s, which corresponds to a capillary number of 10^{-5}.

First, the simulation is performed using standard interFoam. The script runCase.sh run the case for 50 ms with a report interval of 0.05 ms. To run it, simply type:

```
./runCase.sh
```

The script 'runCaseGCFoam.sh' will run the same test case with interGCFoam.

First, check that the 'system/fvSolution' file reads:

```
cSK             0.5;
nSK             1;
cPc             0.1;
\vspace{0.2cm}
```

Now run the following commands:

```
./delete.sh
./runCaseGC.sh
```

The script runCaseGC.sh will perform the simulation with interGCFoam. For both case, the maximum dimensionless velocity (velocity divided by 1.5 $\times$ injection velocity) in the domain is calculated using the processMaxVelocity utility and plotted in Fig. 5. The maximum dimensionless velocity should converge to one. Although the spurious velocities are not eliminated, they are considerably reduced.

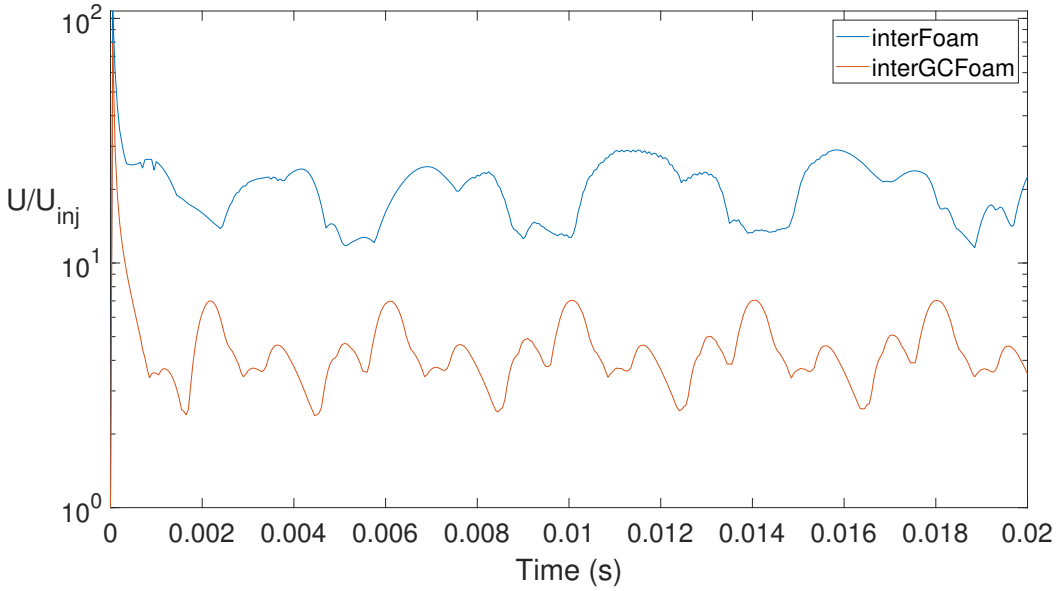

Figure 5: Evolution of the maximum dimensionless velocity during water drainage in oil wet 2D pore throat.

5.5 Bubble in a straight channel with interOSFoam

This test case is an example of the speed-up that can be obtained when using interOSFoam at low capillary number. For this, we consider a 2D straight channel of size 600 μm $\times$ 100 μm. The fluid considered here are air ($\rho_1 = 1$ kg/m^3, $\mu_1 = 18$ μPa.s) and ethanol ($\rho_2 = 789$ kg/m^3, $\mu_2 = 1.2$ mPa.s), and the interfacial tension γ is equal to 20 mN/m. The top and bottom boundaries are wall conditions with fixed contact angle $\theta = 0^o$. The bubble is initialised as a rectangle of length L = 200 μm and width 100 μm, at a position x = 20 μm from the left boundary, and is relaxed until capillary equilibrium. Then, at t=0, we inject from the left boundary ethanol at constant velocity U=0.167 mm/s. The simulation is run until t=0.04 s, at a constant time-step $\Delta t = 4 \times 10^{-5}$ s and a capillary relaxation time-step $\Delta t_{cr} = 5 \times 10^{-8}$ s. To run it, simply type:

```
./runCase.sh
```

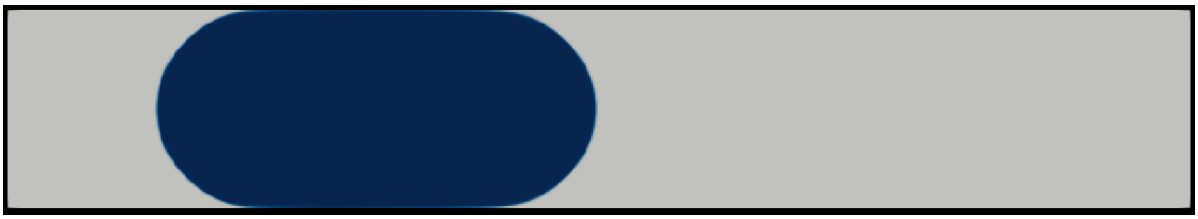

Figure 6: Air bubble in a 2D straight channel at t=0.04 s.

Each time-step corresponds to 800 capillary relaxation steps. However, they converge well before that. The number of relaxation steps can be extracted from the log file using

```
./writeStep.sh
```

and plotted using MATLAB:

The relaxation steps converge in an average of 27, providing a speed-up of approximatively 30$\times$

5.6 2D throat multiphase transport

This test case is an example of how to compute an interface transfer coefficient at phase equilibrium. For this,we consider a 2D microchannel of size 800 μm $\times$ 100 μm with thin film deposition. The fluid considered here are air ($\rho_1 = 1$ kg/m^3, $\mu_1 = 18$ μPa.s) and ethanol ($\rho_2 = 789$ kg/m^3, $\mu_2 = 1.2$ mPa.s), and the interfacial tension γ is equal to 20 mN/m. The top and bottom boundaries are wall conditions with fixed contact angle $\theta = 20^o$.

The domain is initially filled with ethanol and we inject air from the left boundary with a constant velocity $U = 0.4$ m/s while fixing the pressure on the right boundary. These flow parameters correspond to $Ca = 2.4 \times 10^{-2}$. In addition to this, the gas phase carries a species T with a concentration $C = 1$ kg/m^3. The diffusivity of T in the gas and liquid phases is $D_g = D_l = 2 \times 10^{-7}$ m^2/s, which corresponds to $Pe = 100$, and the Henry coefficient H is assumed to be equal to 0.1. The initial concentration of contaminant in the liquid phase is set to zero.

The thickness h of the deposited film on the walls is given by the semi-empirical Taylor's law proposed by [12]

$$\frac{h}{R} = \frac{1.34Ca^{2/3}}{1 + 3.35Ca^{2/3}}, \tag{17}$$

where R is the radius of the microchannel. This corresponds to $h \approx 4.5$ μm. To capture accurately the thin film, we use a grid with constant size $\delta x = 4$ μm in the x-direction and local

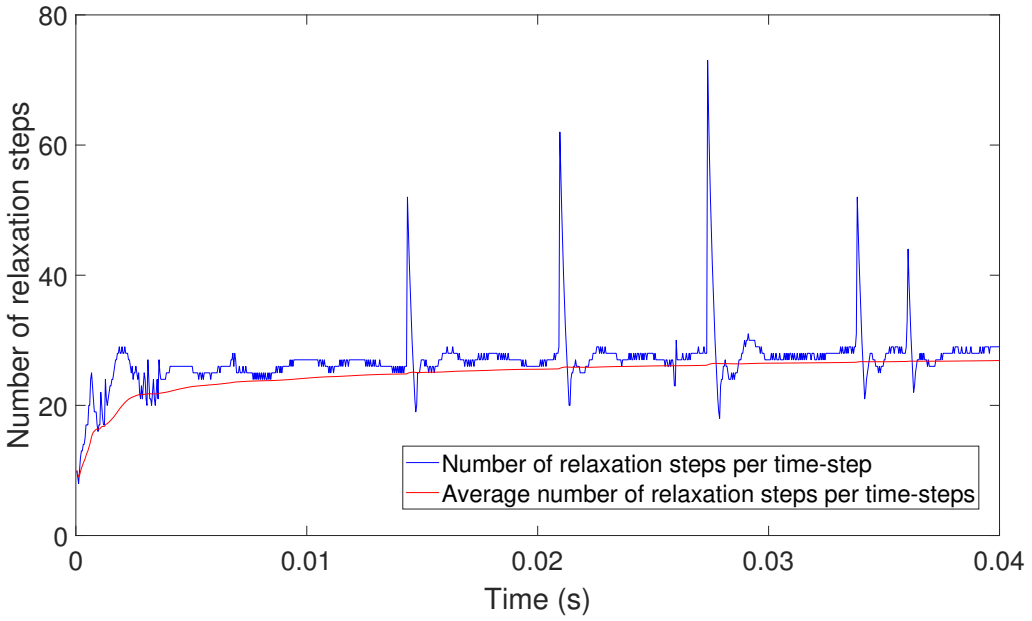

Figure 7: Evolution oif number of relaxation steps per time-step during steady motion of an air bubble in a 2D straight channel at $Ca = 10^{-5}$.

refinement in the y-direction near the wall with $\delta y_{min} = 0.6$ μm on the wall and $\delta y_{max} = 4$ μm at the center of the channel. The full grid includes 200x40 cells. The script runCase.sh runs the case for 2 ms with a report interval of 0.01 ms. To run it, simply type:

```
./runCase.sh
```

The evolution of the phase indicator function and the species concentration can be observed using paraview (see Fig. 8). We obtain a wetting film on the wall with thickness $h = 4.7$ μm.

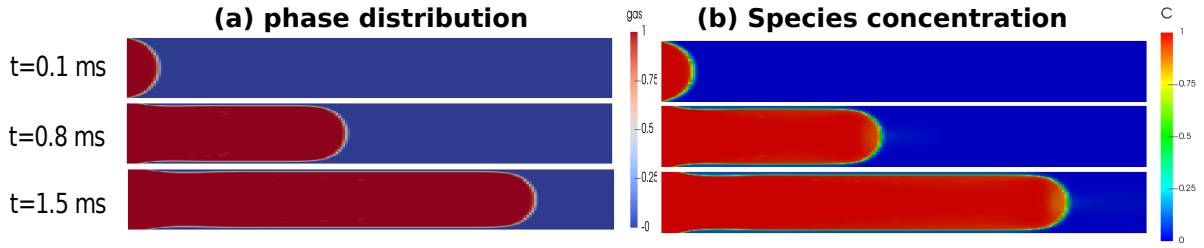

Figure 8: Drainage of ethanol by air in a 2D microchannel. Figure (a): evolution of the phase indicator function. Figure (b): evolution of the species concentration function.

The average concentration in each phase is calculated using the processPhaseConcentration utility. To avoid border effect, the transfer coefficient will be calculated for a truncated domain that goes from x=200 μm to x=600 μm, defined in the 'system/postProcessDict' file. Φ can then be plotted as a function of $\Delta = HC_g - C_l$ from t=0.0004 s (fig. 9).

Next we look more closely at the mass flux per interfacial area from t=1.5 ms 10, which corresponds to the time when the front exit the domain from the right. We observe that the system reach on equilibrium for which the flux is a linear function of the concentration difference Δ:

$$\Phi = \kappa\Delta \tag{18}$$

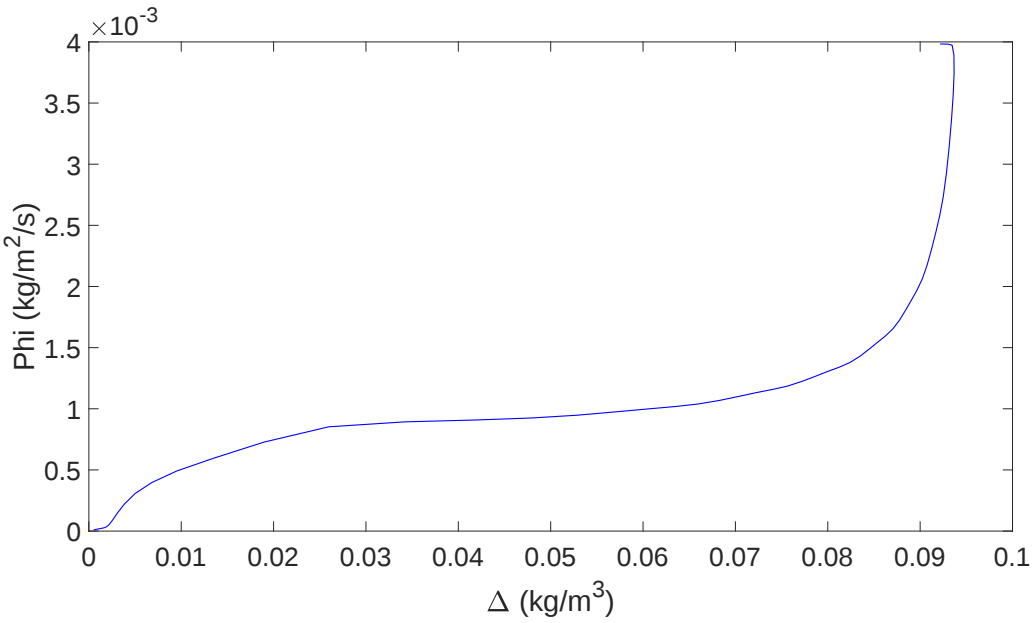

Figure 9: Evolution of the total mass flux per interfacial area Φ as a function of the concentration difference Δ

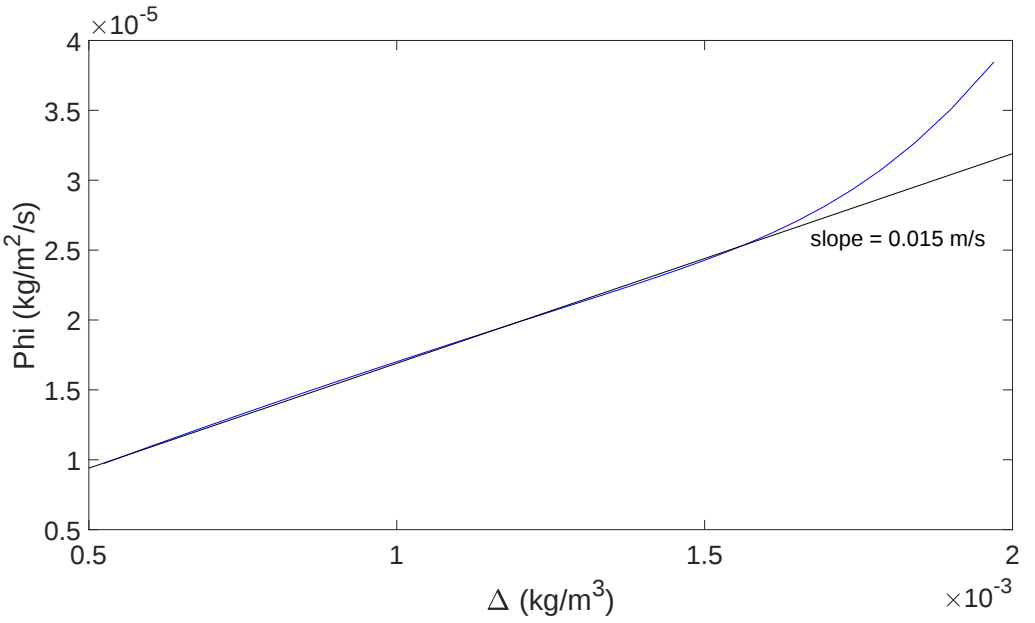

Figure 10: Evolution of the total mass flux per interfacial area Φ as a function of the concentration difference Δ from t=1.5 ms

κ is the mass exchange coefficient. It can be obtain by estimating the slope of the trendline in fig. 10. Here we obtain $\kappa = 0.014\ m/s$

5.7 rising bubble

In this test case, we simulate the dissolution of a rising gas bubble immersed in liquid. We consider a domain of size 1.2 cm × 2.4 cm × 1.2 cm. A gas bubble of radius R = 2 mm is immersed in liquid. The gas is assumed to be pure, and dissolving in water with Henry's constant $H = 2$. The bubble is initially at equilibrium with center positioned at (0 cm, 0.3 cm, 0 cm). As the bubble rises due to buoyancy, interfacial mass transfer occurs and the gas slowly dissolves.

The script runCase.sh runs the case for 0.04 s with a report interval of 0.02 s. To run it, simply type:

```
./runCase.sh
```

Paraview can be used to visualise the evolution of the concentration field (Fig. 11). The colours represent the concentration of gas component and the white line the gas/liquid interface. As it dissolves, the bubble slows down and become more circular. The simulation was stopped at t=0.04 s, but can be run for longer to observe the bubble disappearing.

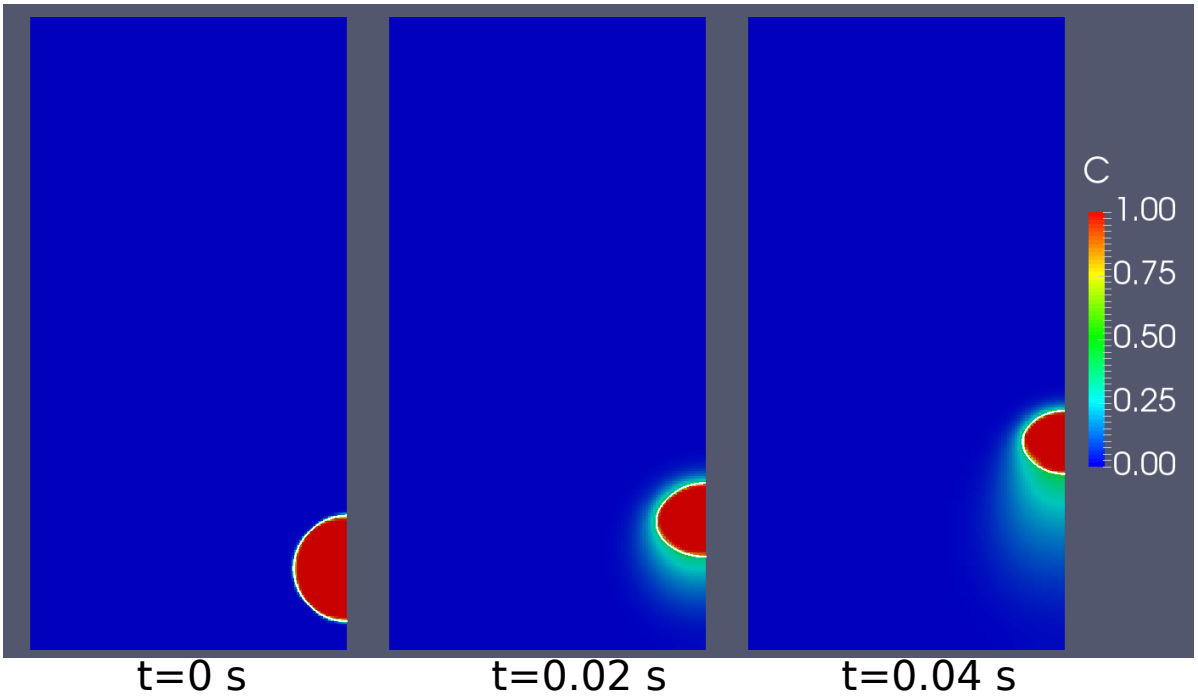

Figure 11: Numerical simulation of dissolution of a rising gas bubble.

5.8 Pore space reactive transport

This test case simulate calcite surface complexation in a oil-filled 2D porous domain. The geometry is defined in 'constant/triSurface' and meshed using the OpenFOAM snappyHexMesh utility.

The domain is a 0.92 mm by 0.28 mm porous media of porosity 0.57. The fluid are oil and water. The solid walls are oil-wet with a contact angle of 135^o. The pore space is initially filled with oil and we inject water from the left boundary at a constant velocity $u = 0.01$ m/s. The surface of the solid has previously been equilibrated with a solution of 1000 mg of $CaCl_2$ in water, and we are injecting a solution with lower concentration of 100 mg of $CaCl_2$. Calcium ions are desorbed from the surface and replaced by hydrogen, which increase the pH near the

pore walls. The script runCase.sh runs the case for 10 ms with a report interval of 0.5 ms. To run it, simply type:

```
./runCase.sh
```

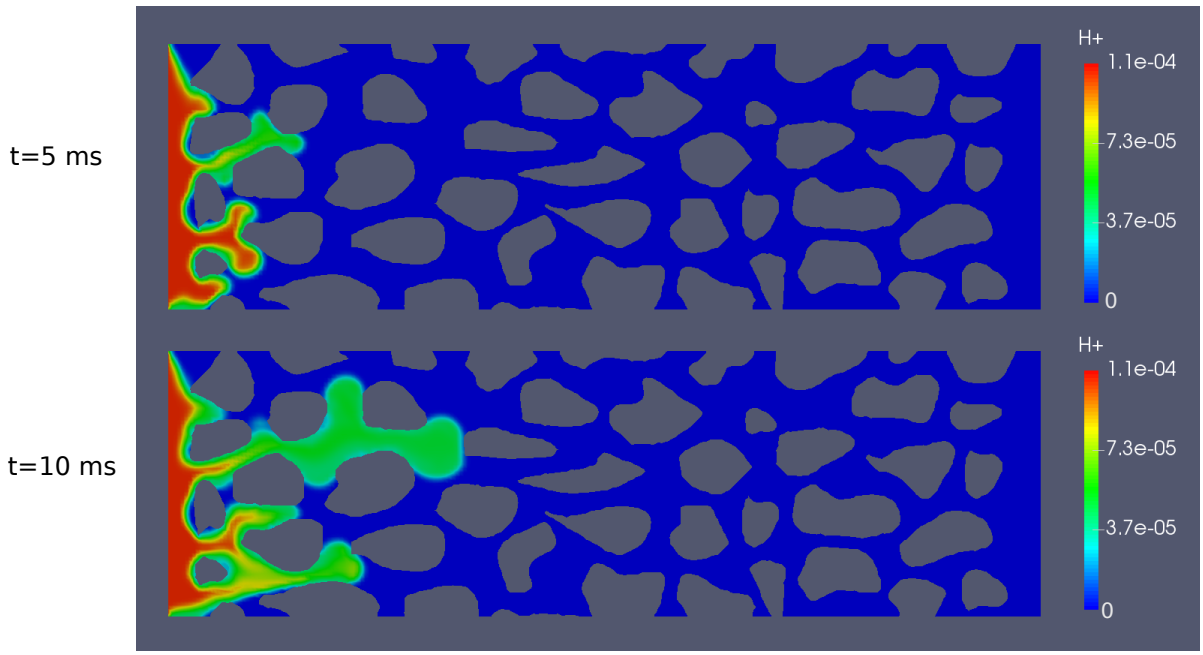

Figure 12: Numerical simulation of multiphase reactive transport in porous media with surface complexation

Fig. 12 shows the evolution of the concentration of H^+ using paraview. As the water front progresses in the domain, more and more Ca^+2 are desorbed from pore walls and replaced by H^+. The pH increases accordingly.

5.9 2DCavity

In this example, we investigate interface transfer and chemical reactions during dissolution of a CO_2 gas bubble in a pore cavity. The geometry is a 6mm×1mm×1mm channel, with a 2mm×2mm×1mm cavity inserted in the middle. The domain is meshed using a 2D uniform grid with resolution 100 microns. Initially, CO_2 gas is trapped in the cavity and the rest is filled with water. The system contains 6 species (H_2O, H^+, OH^-, CO_2, CO_3^{2-} and HCO_3^-). Each species with the exception of H_2O is dilute in the aqueous phase. The gas phase is pure CO_2. At t=0, we inject pure water at pH=7 from the left boundary at a flow rate of 1 mL/min. In order to save on computational time, this example is done with diffusion coefficient two orders of magnitude larger than typical for species in water. The script runCase.sh runs the case for 0.5 s with a report interval of 40 μs. To run it, simply type:

```
./runCase.sh
```

Fig. 13 shows the concentration map of CO_2 and H^+ at t=0.5 s. We see the dissolution of the gas bubble in the water creates a carbonic acid.

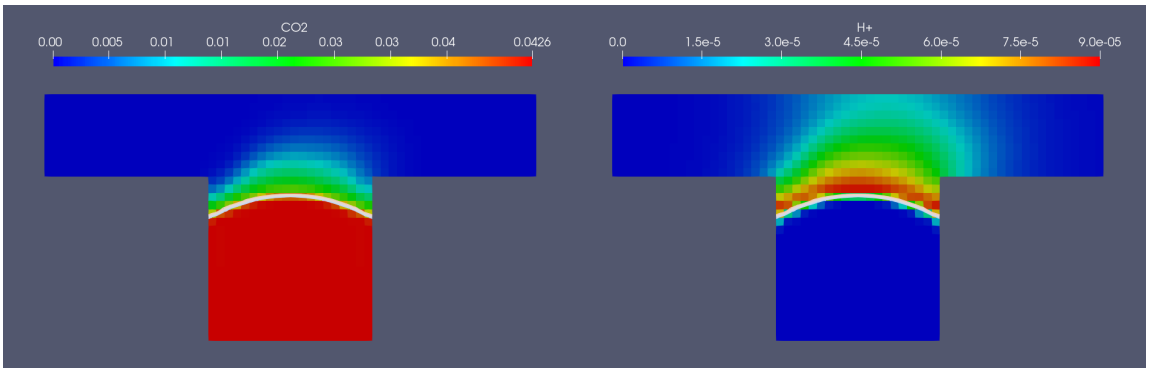

Figure 13: Numerical simulation of multiphase reactive transport during dissolution of a CO_2 gas bubble in a pore cavity

5.10 Estaillades with microporosity

In this test case, simpleDBSFoam is used to calculate the permeability of a 60^3 voxels micro-CT image that includes microporosity (subvolume of estaillades). Microporosity is defined by a porosity lower than 1 and a permeability coefficient. These values are defined in '0/eps' and '0/Kinv' (inverse of permeability).

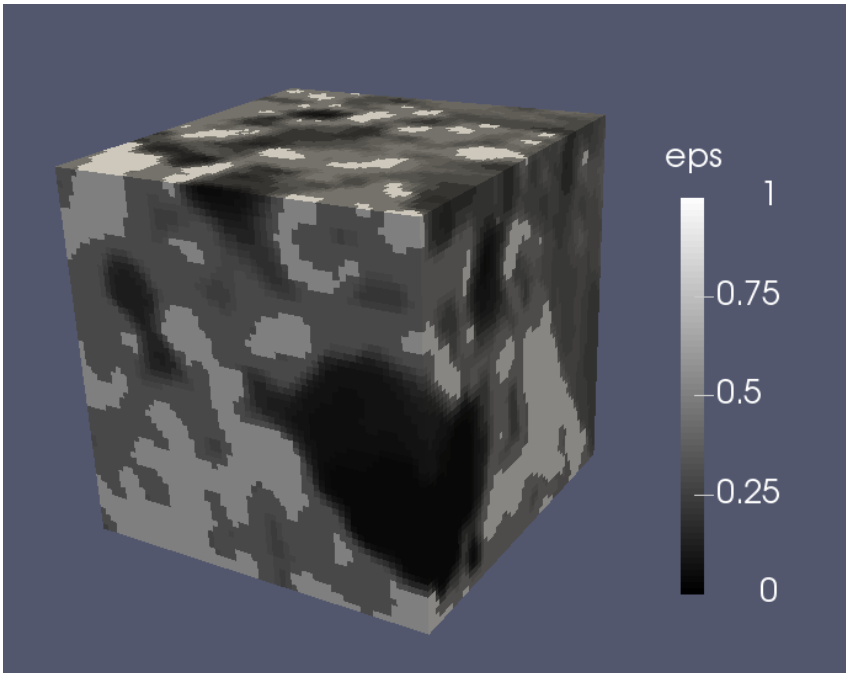

Figure 14: Porosity field for estaillades subvolume.

The 'runCase.sh' script calculate the velocity field and the permeability.

```
./runCase.sh
```

We obtain a permeability of 0.58 Darcy.

5.11 Bentheimer heat transfer

In this test case, supercritical CO_2 is injected through a Bentheimer micro-CT image at a temperature T_0, while the rock is heated at a temperature T_1 represented by constant boundary condition on all side apart form inlet and outlet. The application *heatTransportSimpleFoam* is used to solve the temperature map with convection and conduction in the fluid and conduction in the solid. The domain is meshed using the 'initCaseLMR.sh' script:

```
./initCaseLMR.sh
```

The image is 412×400×400 voxels with a resolution of 5 microns and is located in the 'constant/triSurface' folder, The first 12 voxels in the x direction corresponds to the injection zone. To save on computational time, we consider a maximum resolution of 10 microns for this tutorials. The image is meshed with a two-level grid with local refinement around the solid surface.

Flow and heat transfer are then solved using the 'runcase.sh' script

```
./runCase.sh
```

First, the velocity field is solved using *simpleDBSFoam*, and then the temperature using *heatTransportSimpleFoam*. The new velocity, pressure and temperature are copied in the '0' folder. The temperature in the domain can be observed using paraview.

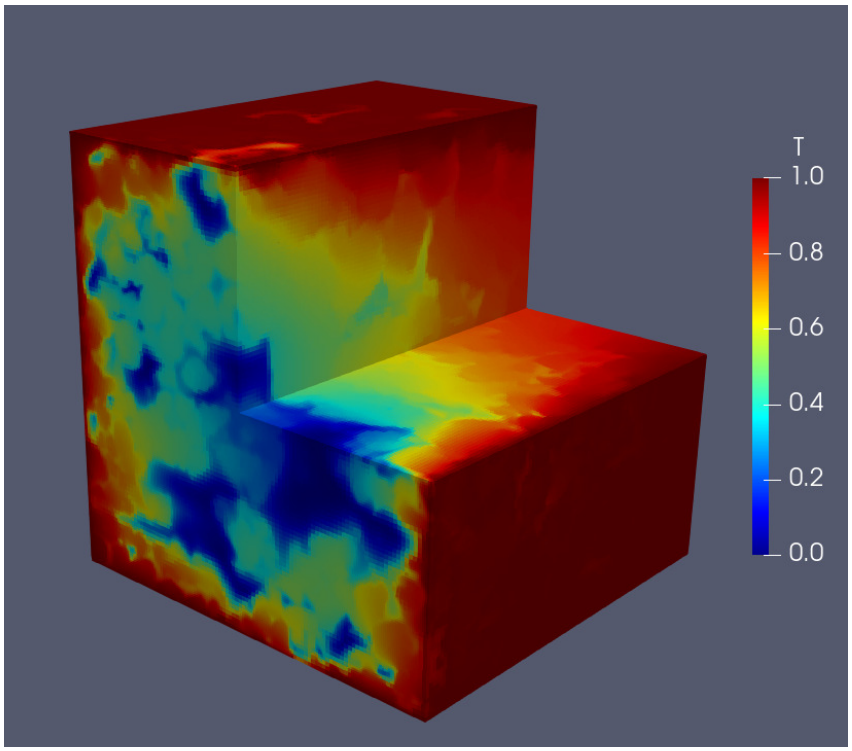

Figure 15: Temperature map during conjugate heat transfer in Bentheimer micro-CT image. The image has been cropped to exclude the injection zone and the temperature is shown on two clips located in the middle of the image in the y and z direction

The 'runCase.sh' script also uses the 'processHeatTransfer' utilities to calculate the heat transfer coefficient between solid and fluid. The coefficient is written in 'HT.csv' and we find 2.6×10^{-3} m/s.

References

[1] Starchenko, V., Marra, C. J., Ladd A. J. C., Three-dimensional simulations of fracture dissolution, J. Geophys. Res. Solid Earth, 2016; 121:6421-6444.

[2] J. U. Brackbill, D. B Kothe, C. Zemach, A continuum method for modeling surface tension, J. Comput. Phys. 1992;2:335-354.

[3] T. Abadie, J. Aubin, D. Legendre, On the combined effects of surface tension force calculation and interface advection on spurious currents within Volume of Fluid and Level Set frameworks, J. Comput. Phys. 2015;297:611–636.

[4] S. Pavuluri, J. Maes, F. Doster, Spontaneous imbibition in a microchannel: analytical solution and assessment of volume of fluid formulations, Microfluidics and Nanofluidics 2018;22:90.

[5] S. Pavuluri, J. Maes, F. Doster, Towards pore network modelling of spontaneous imbibition:contact angle dependent invasion patterns and the occurrence of dynamic capillary barriers, Computational Geosciences 2020; 24:951–969.

[6] J. Maes, Soulaine, C., A new compressive scheme to simulate species transfer across fluid interfaces using the Volume-Of-Fluid method, Chem. Eng. Sci. 2018;190:405-418.

[7] J. Maes, Soulaine, C., A unified single-field Volume-of-Fluid-based formulation for multicomponent interfacial transfer with local volume changes, J. Comput. Phys. 2020; 402 (109024).

[8] M. Graveleau, C. Soulaine, H. Tchelepi., Pore scale simulation of interface multicomponent mass transfer for subsurface flow, Transp. Porous Media 2017;120(2):287-308.

[9] C. Soulaine, F. Gjetvaj, C. Garing, S. Roman, A. Russian, P. Gouze, H. A. Tchelepi, The Impact of Sub-Resolution Porosity of X-ray Microtomography Images on the Permeability, Transp. Porous Med. 2016; 113:227–243.

[10] C. Soulaine, S. Roman, A. Kovscek, H. A. Tchelepi,Mineral dissolution and wormholing from a pore-scale perspective,J. Fluid Mech.2017, 827:457-483.

[11] L. Talon, D. Bauer, N. Gland, S. Youssef, H. Auradou, I. Ginzburg, Assessment of the two relaxation time Lattice-Boltzman scheme to simulate Stokes flow in porous media. Water Resour. Res. 2012; 48(4);W07406

[12] P. Aussilous, D. Quere, Quick deposition of a fluid on the wall of a tube, Phys. Fluids, 2000; 12:2367-2371.

[13] J. Maes, H. P. Menke, GeoChemFoam: Direct Modelling of Multiphase Reactive Transport in Real Pore Geometries with Equilibrium Reactions, Transp. Porous Med., 139:271-299, 2021

[14] J. Maes, H. P. Menke, GeoChemFoam: Operator Splitting based time-stepping for efficient Volume-Of-Fluid simulation of capillary-dominated two-phase flow, arXiv:2105.10576, 2021

[15] J. Maes, H. P. Menke, GeoChemFoam: Direct Modelling of flow and heat transfer in micro-CT images of porous media, arXiv:2110.03311, 2021